IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Appellants:	Daniel N. CRIPE et al.	§	Confirmation No.:	8712
Serial No.:	10/717,730	§	Group Art Unit:	2616
Filed:	11/20/2003	§	Examiner:	P. Sinkantarakorn
For:	Method And System Of Teamed Network Adapters With Offloaded Connections	§	Docket No.:	200313587-1

APPEAL BRIEF

Mail Stop Appeal Brief – Patents
Commissioner for Patents
PO Box 1450
Alexandria, VA 22313-1450

Date: October 6, 2008

Sir:

Appellants hereby submit this Appeal Brief in connection with the above-identified application. A Notice of Appeal was electronically filed on August 5, 2008.

TABLE OF CONTENTS

I. REAL PARTY IN INTEREST 3
II. RELATED APPEALS AND INTERFERENCES 4
III. STATUS OF THE CLAIMS 5
IV. STATUS OF AMENDMENT 6
V. SUMMARY OF THE CLAIMED SUBJECT MATTER 7
VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL 9
A. Summary of Burns 10
B. Rejection Under 35 U.S.C. § 103(a) in View of Congdon and Burns – Claims 1-4 and 6-21 10
C. Rejection Under 35 U.S.C. § 103(a) in View of Congdon, Burns and Mahalingham - Claim 5 12
D. Conclusion 12
VII. CLAIMS APPENDIX 14
VIII. EVIDENCE APPENDIX 18
IX. RELATED PROCEEDINGS APPENDIX 19

I. REAL PARTY IN INTEREST

The real party in interest is the Hewlett-Packard Development Company, L.P. (HPDC), a Texas Limited Partnership, having its principal place of business in Houston, Texas. HPDC is a wholly-owned affiliate of Hewlett-Packard Company (HPC). The Assignment from the inventors to HPDC was recorded on November 20, 2003 at Reel/Frame 014724/0976.

II. RELATED APPEALS AND INTERFERENCES

Appellants are unaware of any related appeals or interferences.

III. STATUS OF THE CLAIMS

Originally filed claims:	1-21.
Claim cancellations:	None.
Added claims:	None.
Presently pending claims:	1-21.
Presently appealed claims:	1-21.

IV. STATUS OF AMENDMENT

Appellants have not filed any amendments after the Final Office Action dated May 16, 2008.

V. SUMMARY OF THE CLAIMED SUBJECT MATTER

Various embodiments of the invention are described below. The scope of disclosure is not limited by the descriptions of the embodiments that follow. Citations to the specification have been provided to demonstrate where support may be found in the specification for various parts of the invention. Additional support may be found elsewhere in the application.

Appellants' contribution is directed to a technique for offloading and reloading network connections among multiple Network Interface Cards (NICs) 118, 120, 122 that have been teamed together. Specifically, a server 102 may communicate with one or more clients 104, 106 via a network switch 124. P. 3, ll. 21-23; Figure 1. The server 102 communicates with the network switch 124 using a plurality of NICs 118, 120, 122. P. 4, ll. 1-2; Figure 1. Processing of a TCP/IP stack 114 (used to establish and maintain network connections) in the server 102 is performed by a CPU 108, but may be "offloaded" (or "moved") to circuit logic in one of the NICs 118, 120, 122. P. 5, l. 31 – p. 6, l. 5; Figures 1-2. When NICs are operating in a team, they share common addresses. P. 6, ll. 22-24; Figure 1. Thus, packets received by the team may be received by any one of the NICs in the team. P. 3, ll. 30-31; Figure 1. When a NIC in the team becomes inoperative (*e.g.*, due to failure), another NIC in the team may receive the packet. P. 8, ll. 4-8; Figure 1. The technique mentioned above comprises detecting the receipt of a packet on a different-than-usual NIC and, as a result, offloading a network connection from the defective NIC to the NIC on which the packet was received. P. 9, ll. 8-16; Figures 1 and 4. Clearly, therefore, this offloading from the defective NIC to the NIC on which the packet was received **is precipitated by the receipt of the packet on that non-defective NIC**.

Claim 1 is directed to a computer system 100 that comprises a central processing unit (CPU) 108 and first and second network adapters 118, 120, 122 teamed together and configured to receive offloaded connections. P. 5, l. 31 – p. 6, l. 5; p. 6, ll. 22-24; Figure 1. A program 116 executing on the CPU 108 reloads an offloaded connection established by the first network adapter 118, 120, 122 onto the second network adapter 118, 120, 122 as a result of one of a

plurality of packets associated with the offloaded connection being received on the second network adapter 118, 120, 122. P. 9, ll. 8-16; Figures 1 and 4.

Claim 8 is directed to a method that comprises examining a packet received from an external device 124 and determining whether a connection associated with the packet is currently offloaded. Col. 8, l. 29 – col. 9, l. 3; Figures 1 and 4. The method also comprises reloading the connection in response to the packet associated with the connection being offloaded and received by a network interface 118, 120, 122 not currently processing the offloaded connection. P. 9, ll. 8-16; Figures 1 and 4.

Claim 12 is directed to a computer readable media 110 storing instructions 116 executable by a computer system 100, and when executed the instructions implement a method that comprises examining a packet received from an external device 124 and determining whether a connection 118, 120, 122 associated with the packet is currently offloaded. Col. 8, l. 29 – col. 9, l. 3; Figures 1 and 4. The method also comprises reloading the connection as a result of the packet associated with the connection being offloaded and received by a network interface 118, 120, 122 not currently processing the offloaded connection. P. 9, ll. 8-16; Figures 1 and 4.

Claim 16 is directed to a computer system 100 comprising means (*e.g.*, CPU 108; Figure 1) for reading and executing programs. Figure 1. The system 100 also comprises first and second means (*e.g.*, NICs 118, 120, 122; Figure 1) for sending and receiving data connections over a network, where the first and second means are grouped together and are capable of processing offloaded data connections. P. 3, ll. 21-23; p. 5, l. 31 – p. 6, l. 5; Figure 1. A program executed by the means for reading and executing programs reloads an offloaded connection established by the first means for sending and receiving data onto the second means for sending and receiving data in response to one of a plurality of packets associated with the offloaded connection being received on the second means for sending and receiving data. P. 9, ll. 8-16; Figures 1 and 4.

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

Whether under 35 U.S.C. § 103(a) claims 1-4 and 6-21 are obvious in view of Congdon (U.S. Pat. No. 6,151,297) and Burns (U.S. Pat. No. 6,938,092).

Whether under 35 U.S.C. § 103(a) claim 5 is obvious in view of Congdon, Burns and Mahalingham (U.S. Pat. No. 6,314,525).

VII. ARGUMENT

A. Summary of Burns

The Examiner cites two portions of Burns as being relevant: col. 6, ll. 36-67 and col. 7, ll. 57-67. Appellants describe each portion in turn.

Col. 6, ll. 36-67 relate to Burns, Figure 2. Figure 2 shows a Network Interface Device (NID) 103 that sends and receives data to and from a hub/switch 105 via ports 1 and 2. Port 1 transfers data from the hub/switch 105 to NID 103; port 2 transfers data from the NID 103 to the hub/switch 105. Port 1 is associated with MAC address "A," while port 2 is associated with MAC address "C." Both ports 1 and 2 are said to be associated with "connection #1." Burns teaches that when a failure occurs on Port 1, the NID 103 detects the failure and notifies a port aggregation driver 112. In turn, the driver 112 updates its data structures accordingly. Ports 1 and 2 are part of the same team. Port 1 may be designated as the primary port. When port 1 fails, the driver 112 chooses port 2 to be the new, primary team member by adjusting a pointer as shown in Figure 3.

Despite these teachings in col. 6, ll. 36-67, Burns does not teach that port 2 is selected as a result of receiving a packet associated with the connection previously coupled to port 1.

Col. 7, ll. 57-67 further describes what occurs if port 1 fails. In particular, if port 2 is to continue transmitting for connection #1, and if port 2's MAC address is to be changed from MAC C to MAC A, the driver 112 updates the TCB 119 for connection #1 by changing the MAC source address. **Again, however, Burns does not teach or even suggest that port 2's MAC address is changed or updated as a result of receiving a packet associated with the connection previously coupled to port 1.**

B. Rejection Under 35 U.S.C. § 103(a) in View of Congdon and Burns – Claims 1-4 and 6-21

The Examiner rejected claims 1-4 and 6-21 as allegedly obvious in view of Congdon and Burns. Appellants traverse this rejection. Independent claim 1 is representative of this grouping of claims. The grouping should not be construed to mean the patentability of any of the claims may be determined in later actions

(*e.g.*, actions before a court) based on the groupings. Rather, the presumption of 35 USC § 282 shall apply to each of these claims individually.

Claim 1 requires "wherein a program executing on the CPU reloads an offloaded connection established by the first network adapter onto the second network adapter **as a result of one of a plurality of packets associated with the offloaded connection being received on the second network adapter**" (emphasis added). The Examiner admits that although Congdon teaches fault tolerance, Congdon fails to teach this limitation in its entirety. As a result, the Examiner turns to Burns and asserts that Burns, combined with Congdon, teaches this claim limitation.

Respectfully, the Examiner is mistaken. The cited portions of Burns, described above in Section VII(A) of this Brief, fail to teach or suggest this limitation, and combining fault tolerance as taught by Congdon fails to satisfy Burns' deficiencies. First, the claim limitation above requires that the receipt of "one of a plurality of packets associated with the offloaded connection" on "[a] second network adapter" **be the cause** for "reload[ing] an offloaded connection" of the first network adapter to the second network adapter. Nowhere does Burns appear to teach that the transfer of a connection between ports 1 and 2 occurs **due to the receipt of one of a plurality of packets associated with the offloaded connection on either of these ports**. In fact, Burns does not appear to teach the transfer of connections at all – Burns only teaches the alteration of a MAC address on a surviving port (*e.g.*, port 2, col. 7, l. 57). Second, the cited portions of Burns discuss inactivation of port 1 (see Burns, Fig. 2). Because only port 2 remains active, it is not possible for any packets to be received at all, much less for offloaded connections to be transferred between network adapters as a result of such packets being received. Although Congdon appears to teach fault tolerance, fault tolerance alone does not satisfy Burns' deficiencies, highlighted above. Thus, the combination of Burns and Congdon cannot and does not teach or suggest all limitations of claim 1.

The claim limitation argued above is significant at least because it directly pertains to how Appellants' contribution achieves one of its objectives.

Specifically, referring to Fig. 1 of Appellants' application, when one of the NICs 118, 120, 122 fails or otherwise becomes inactive, another NIC with which the failed NIC is teamed may begin receiving packets from the network switch 124 that would otherwise have been received by the failed NIC. P. 7, ll. 5-6. This may occur because network switch 124 has detected failure of the failed NIC and has re-routed its packets to go to another NIC in the same team as the failed NIC. P. 7, ll. 5-6. In order for the NICs to continue operating as a team, the server 102 detects the NIC on which the packets (packets that would originally have gone to the failed NIC) are received. P. 8, l. 9 – p. 9, l. 16; Figures 1 and 4. The server 102 re-loads or offloads the connection of the failed NIC to the NIC on which these packets were received. P. 8, l. 9 – p. 9, l. 16; Figures 1 and 4. Thus, the limitation "wherein a program executing on the CPU reloads an offloaded connection established by the first network adapter onto the second network adapter **as a result of** one of a plurality of packets associated with the offloaded connection being received on the second network adapter" (claim 1; emphasis added) is significant at least because it pertains directly to how the server 102 achieves its objective.

Based on the foregoing, the Examiner erred in rejecting claims 1-4 and 6-20. Appellants respectfully request that the claims in this grouping be set for issue.

C. Rejection Under 35 U.S.C. § 103(a) in View of Congdon, Burns and Mahalingham - Claim 5

The Examiner rejected claim 5 as allegedly obvious in view of Congdon, Burns and Mahalingham. Appellants traverse this rejection. As explained above, the Examiner erred in rejecting claim 1 using the combination of Congdon and Burns. Mahalingham fails to satisfy the deficiencies of this combination. Thus, the Examiner erred in rejecting claim 1 and all claims dependent on claim 1 (including claim 5) using the combination of Congdon, Burns and Mahalingham.

D. Conclusion

For the reasons stated above, Appellants respectfully submit that the Examiner erred in rejecting all pending claims. It is believed that no extensions

of time or fees are required, beyond those that may otherwise be provided for in documents accompanying this paper. However, in the event that additional extensions of time are necessary to allow consideration of this paper, such extensions are hereby petitioned under 37 C.F.R. § 1.136(a), and any fees required (including fees for net addition of claims) are hereby authorized to be charged to Hewlett-Packard Development Company's Deposit Account No. 08-2025.

Respectfully submitted,

/Nick P. Patel/

Nick P. Patel
PTO Reg. No. 57,365
CONLEY ROSE, P.C.
(713) 238-8000 (Phone)
(713) 238-8008 (Fax)
AGENT FOR APPELLANTS

HEWLETT-PACKARD COMPANY
Intellectual Property Administration
Legal Dept., M/S 35
P.O. Box 272400
Fort Collins, CO 80527-2400

VII. CLAIMS APPENDIX

1. (Previously presented) A computer system comprising:

a central processing unit (CPU); and

a first and second network adapter teamed together and configured to receive offloaded connections;

wherein a program executing on the CPU reloads an offloaded connection established by the first network adapter onto the second network adapter as a result of one of a plurality of packets associated with the offloaded connection being received on the second network adapter.

2. (Original) The system of claim 1 wherein the first and second network adapters are capable of fully offloading all protocol processing.

3. (Original) The system of claim 1 wherein the first and second network adapters transmit and receive packets of data using a single media access control (MAC) and internet protocol (IP) address.

4. (Original) The system of claim 1 wherein the program reloads an offloaded connection by transferring the context of the connection from the first network adapter to the second network adapter.

5. (Original) The system of claim 1 wherein the program monitors every packet received by the first and second network adapters and inactivates connections associated with packets that have not been received for a defined time period.

6. (Original) The system of claim 1 wherein the first and second network adapters send a notification to the program if a connection is prematurely terminated.

7. (Original) The system of claim 1 wherein the first and second network adapters comprise network interface cards (NICs).

8. (Previously presented) A method comprising:

examining a packet received from an external device;

determining whether a connection associated with the packet is currently offloaded; and

reloading the connection in response to the packet associated with the connection being offloaded and received by a network interface not currently processing the offloaded connection.

9. (Original) The method of claim 8 further comprising determining an identifier for the network interface that receives the packet and writing the determined identifier to a memory.

10. (Original) The method of claim 8 wherein the reloading further comprises copying the context of the connection to the network interface that received the packet.

11. (Original) The method of claim 8 wherein the network interface that received the packet and the network interface currently offloading the connection are teamed together.

12. (Previously presented) A computer readable media storing instructions executable by a computer system, and when executed the instructions implement a method comprising:

examining a packet received from an external device;

determining whether a connection associated with the packet is currently offloaded; and

reloading the connection as a result of the packet associated with the connection being offloaded and received by a network interface not currently processing the offloaded connection.

13. (Original) The computer readable media of claim 12 further comprising determining an identifier for the network interface that receives the packet and writing the determined identifier to a memory unit.

14. (Original) The computer readable media of claim 12 wherein the reloading further comprises copying the context of the connection to the network interface that received the packet.

15. (Original) The computer readable media of claim 12 wherein the network interface that received the packet and the network interface currently offloading the connection are teamed together.

16. (Previously presented) A computer system comprising:

means for reading and executing programs; and

first and second means for sending and receiving data connections over a network, the first and second means grouped together and capable of processing offloaded data connections;

wherein a program executed by the means for reading and executing programs reloads an offloaded connection established by the first means for sending and receiving data onto the second means for sending and receiving data in response to one of a plurality of packets associated with the offloaded connection being received on the second means for sending and receiving data.

17. (Original) The system of claim 16 wherein the first and second means for sending and receiving data connections are capable of fully offloading all protocol processing.

18. (Original) The system of claim 16 wherein the first and second means for sending and receiving data connections send and receive packets of data using a single media access control (MAC) and internet protocol (IP) address.

19. (Original) The system of claim 16 wherein the program reloads an offloaded connection by transferring the context of the connection from the first means for sending and receiving data connections to the second means for sending and receiving data connections.

20. (Original) The system of claim 16 wherein the program monitors all data received by the first and second means for sending and receiving data connections.

21. (Original) The system of claim 16 wherein the first and second means for sending and receiving data connections send a notification to the program if a connection is prematurely terminated.

VIII. EVIDENCE APPENDIX

None.

IX. RELATED PROCEEDINGS APPENDIX

None.